Available online at www.sciencedirect.com
SCIENCE DIRECT®
Comput. Methods Appl. Mech. Engrg. 192 (2003) 2649–2667
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Polyhedrization of an arbitrary 3D point set

Sergio R. Idelsohn [a,b,*], Nestor Calvo [a], Eugenio Oñate [b]

[a] International Center for Computational Methods in Engineering (CIMEC-INTEC), Universidad Nacional del Litoral and CONICET, 3000 Santa Fe, Argentina
[b] International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña, 088034 Barcelona, Spain

Received 30 December 2002; received in revised form 17 March 2003

Abstract

Given a 3D point set, the problem of defining the volume associated, dividing it into a set of regions (elements) and defining a boundary surface is tackled.

Several physical problems need to define volume domains, boundary surfaces and approximating functions from a given point distribution. This is for instance the case of particle methods, in which all the information is the particle positions and there are not boundary surfaces definition.

Until recently, all the FEM mesh generators were limited to the generation of simple elements as tetrahedral or hexahedral elements (or triangular and quadrangular in 2D problems). The reason of this limitation was the lack of any acceptable shape function to be used in other kind of geometrical elements. Nowadays, there are several acceptable shape functions for a very large class of polyhedra. These new shape functions, together with a generalization of the Delaunay tessellation presented in this paper, gives an optimal marriage and a powerful tool to solve a large variety of physical problems by numerical methods.

The domain partition into polyhedra presented here is not a standard mesh generation. The problem here is: for a given node distribution to find a suitable boundary surface and a suitable mesh to be used in the solution of a physical problem by a numerical method. To include new nodes or change their positions is not allowed.

Keywords: Polyhedral mesh generation; Particles methods; Lagrangian formulations; Delaunay; Voronoï

1. Introduction

Several numerical methods in computational mechanics, as well as other volume integration methods need to subdivide the total domain in subdomains called "*elements*". This is the case for the finite element

* Corresponding author. Address: International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña, Barcelona 088034, Spain.
E-mail addresses: sergio@ceride.gov.ar (S.R. Idelsohn), onate@cimne.upc.es (E. Oñate).

0045-7825/03/$ - see front matter
doi:10.1016/S0045-7825(03)00298-6

method (FEM) and the finite volume method. This subdivision is called "*the mesh*" and, in order to be used in a numerical solution of a physical problem, it must satisfy several constraints.

The standard mesh generation problem starts defining the domain by means of the boundary surface (usually a CAD definition) and then:

(a) Divide the total domain into elements by an advancing front technique (AFT) [1], or
(b) Introduce a point distribution in the domain and then perform the partition via a Delaunay tessellation (DT) [2].

Mesh generators have an optimization stage (sometimes called *smoothing* or *cosmetic process*), which is an iterative algorithm designed to improve element shapes until an acceptable partition is reached [3,4].

Both methods have advantages and disadvantages, which may be summarized as:

(a) In the AFT, the boundary surface elements are easily defined, but the method is more expensive compared with the DT. For this reason in problems with permanent mesh update is not recommended.
(b) The big advantage of the DT is its low computational cost. Unfortunately a cosmetic process is necessary in order to use the mesh in a numerical method. Furthermore, the original boundary (CAD) must be recovered [5].

In this paper, the starting problem is quite different from the standard mesh generation one. The difficulty of generating a mesh with a perfect match to the given CAD surface is not an issue here. In this paper, the problem is: *Given a node distribution, find an acceptable mesh and acceptable boundary surface in order to solve some differential equation with some appropriate boundary conditions.*

This is the permanent problem found in particle methods [6–9] or in any fluid mechanics problem solved using a Lagrangian formulation [10,11]. In these problems, at each time step a new node distribution is found and the boundary surfaces and the mesh must be updated or re-generated in order to solve again the equations.

Several other problems as volume reconstruction or image recognition from scanned points may also be solved by the method proposed in this paper [12]. This method can also be used as a tool in any meshless method, where the local cloud identification is a permanent source of trouble [13–17].

The goal of this paper is to obtain from an arbitrary node distribution a partition of the total domain. This partition will be optimal in order to be used in a numerical method as the FEM and the number of operations to obtain this partition must be bounded with n^{β}, where n is the total number of nodes and β must be near the unity. Furthermore, the partition will be unique.

2. The Delaunay tessellation

In order to better understand the new procedure, classical definitions will be introduced for some entities: Voronoï diagrams, DTs, Voronoï spheres, and natural neighborhood

Let a set of distinct nodes be: $\mathbf{N} = \{\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3, \ldots, \mathbf{n}_n\}$ in $\mathbb{R}^3$.

(a) The *Voronoï diagram* of the set $\mathbf{N}$ is a partition of $\mathbb{R}^3$ into convex regions V_i (closed or unbounded), where each region V_i is associated with a node $\mathbf{n}_i$, such that any point in V_i is closer to $\mathbf{n}_i$ (nearest neighbor) than to any other node $\mathbf{n}_k$. See Fig. 1 for a 2D representation. There is a single Voronoï diagram for each set $\mathbf{N}$.
(b) A *Voronoï sphere* within the set $\mathbf{N}$ is any sphere, defined by four or more nodes without any node inside. Such spheres are also known as empty circumspheres.

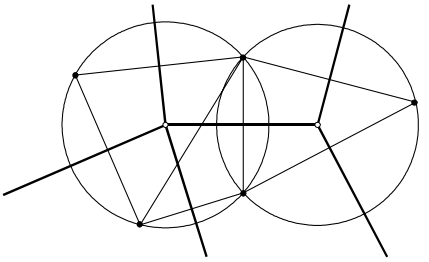

Fig. 1. Voronoï diagram, Voronoï circle and Delaunay triangulation in 2D.

(c) The DT within the set $\mathbf{N}$ is a partition of the convex hull Ω of all the nodes into regions Ω_i such that $\Omega = \cup\Omega_i$, where each Ω_i is a tetrahedron defined by four nodes of the same Voronoï sphere. DTs of a set $\mathbf{N}$ are not unique, but each tessellation is dual to the single Voronoï diagram of the set.
(d) The *natural neighborhood* of any point $\mathbf{x}$ is the region made by the union of the tetrahedra defined by the Voronoï spheres containing the node, its limiting nodes are referred as the *natural neighbors* of the point. These nodes have positive weights on some interpolants over the point $\mathbf{x}$ [18].

The computing time required for the evaluation of all these three entities, by using the Bowyer/Watson algorithm [19,20] with randomized insertion is $O(n^{1+1/D})$ and by means of an octree organization it is bounded by $O(n\log(n))$ [21,22]. In practice the cost is very close to $O(n)$.

So defined, the DT of a set of nodes is non-unique. For the same node distribution, different tetrahedrations are possible. This means that a partition based on the DT is very sensitive to small perturbations of the node positions (see Fig. 2). On the other hand, its dual, the Voronoï diagram and also the Voronoï spheres and natural neighborhoods are unique. Thus, it makes more sense to define a partition based on the unique Voronoï spheres instead of the DT. In Fig. 2, two critical cases of Delaunay instabilities are represented. One is the case of four nodes on the same circle and the other is the case of a node close to a boundary. In both cases, the Voronoï diagram and the Voronoï spheres remains almost unchanged.

Furthermore, in 3D problems the DT may generate several tetrahedra of zero or almost zero volume [23], introducing large inaccuracies into the shape function derivatives. This is the reason why a DT must be iteratively improved in order to obtain an acceptable partition to be used in a numerical method. This

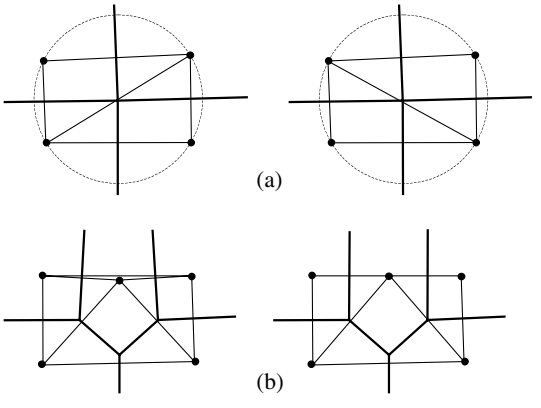

Fig. 2. Instabilities on the DT. (a) Four nodes on the same circle; (b) node close to a boundary.

iterative process is expensive when the tessellation must be permanently updated. The extended Delaunay tessellation (EDT) described below will solve this handicap.

3. The extended Delaunay tessellation

From the point of view of the application of a domain partition to the solution of a numerical method, the best partition is that which the elements have:

(a) all its nodes on the same empty sphere (related to the concept of natural neighbor nodes), and
(b) the polyhedral elements fill the sphere as much as possible (related to the concept of optimal angle between faces).

The standard DT satisfies both previous statements only for 2D problems. Unfortunately, the second statement is not necessarily satisfied in a 3D partition.

The drawbacks appear in the so-called "degenerated case", which is the case where more than four nodes (or more than three nodes in a 2D problem) are on the same empty sphere. For instance, in 2D, when four nodes are on the same circumference, two different triangulations satisfy the Delaunay criterion. However, the most dangerous case appears only in 3D. For instance, when five nodes are on the same sphere, five tetrahedra may be defined satisfying the Delaunay criterion, but some of them may have zero or almost zero volumes, called *slivers* (see Fig. 3).

In order to overcome the drawbacks referred to in the above paragraphs, a generalization of the DT will be defined and termed *EDT*.

Definition. The EDT within the node set **N** is the unique partition of the convex hull of all the nodes into regions Ω_i such that $\Omega = \cup\Omega_i$, where each Ω_i is the polyhedron defined by all the nodes laying on nearby Voronoï spheres. Any two spheres will be considered as "nearby" if their centers are closer than a predefined threshold value δ (see Appendix A).

The main difference between the traditional DT and the EDT is that, in the latter, all the nodes belonging to the same and nearby Voronoï sphere define a unique polyhedron. With this definition, the domain will be divided into tetrahedra and other polyhedra, which are unique for a given δ parameter, satisfying then one of the goals required in Section 1.

Fig. 4 for instance, is a 2D polygon partition with a triangle, a quadrangle and a pentagon. Fig. 5 is a polyhedron with all the nodes on the same sphere, which may appear in a 3D problem.

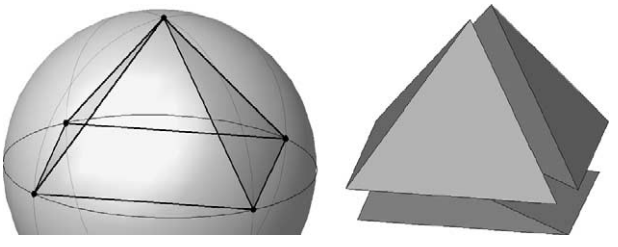

Fig. 3. Five nodes on the same sphere and possible partition with a zero volume sliver on the right.

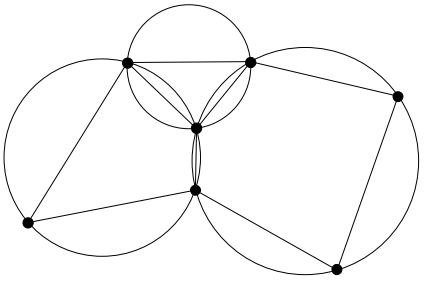

Fig. 4. Two-dimensional partition in polygons. The triangle, the quadrangle and the pentagon are each inscribed on a circle.

Fig. 5. Eight-node polyhedron. All nodes are on the same sphere.

When δ grows, the number of polyhedra with more than four nodes will increase, and the number of tetrahedra with near-zero volume will decrease, and vice versa.

Then, the role of the EDT is to provide the tetrahedral partition when more than four nodes are on the same empty sphere (or near the same). When this is the case, *the best element to be used in a numerical method is the polyhedron formed with all these nodes*.

The implementation was made starting with a standard DT using the Bowyer/Watson algorithm mentioned above. Then the similar spheres were joined (Appendix A) in a node-wise search guaranteeing the $O(n)$ of the algorithm.

The Bowyer/Watson algorithm was chosen for its easy implementation with the same structure both in 2D and 3D and mainly due to its rely on the Voronoï spheres. However, some care must be taken with numerical precision problems when there are more than four cospherical nodes [21,24,25]. Nevertheless, the joining similar spheres has no such problems. When the δ value used is far greater than the numerical precision, each sphere captures all the cospherical and nearby nodes.

The EDT allows the existence of a domain partition which: (a) is unique for a given δ, (b) does not have polyhedra with near-zero volume, and (c) is obtained in a bounded time of the same order as the DT (order $\approx n$). Then, it satisfies all the goals stated previously.

It must be noted that until quite recently, all the mesh generators were limited to the generation of very simple shapes, mainly tetrahedral [12] or hexahedral [26]. The reason of this limitation was the lack of any acceptable shape function to be used in other kind of geometrical elements. Nowadays, several acceptable shape functions for a larger family of polyhedra are easily generated, particularly for polyhedra having its vertices on a spherical surface (see for instance Refs. [27,28] or Appendix B). These new shape functions, together with the EDT gives an optimal marriage and a powerful tool to solve a large variety of physical problems by numerical methods.

4. The boundary surface

As stated before one of the problems in standard Delaunay mesh generation is the correct match of the mesh boundary with the previously defined surfaces. The problem here is quite easier. There are not previously defined surfaces, there are only node positions and then, the definition of the boundary surface is not unique.

Sometimes, boundary nodes are explicitly defined as special nodes, which are different from internal nodes. In other cases, the total set of nodes is the only information available and the algorithm must recognize the boundaries. Such is the case for instance, for the Lagrangian formulation in fluid mechanics problems [10,11] in which, at each time step, a new node distribution is obtained and the free surface must be recognized from the node positions.

The use of Voronoï spheres may make it easier to recognize boundary nodes. By considering an arbitrary node distribution, with $h(\mathbf{x})$ as the typical distance between neighboring nodes, the following criterion will be used:

All nodes defining an empty sphere with a radius $r(\mathbf{x})$ larger than $\alpha h(\mathbf{x})$, are considered as boundary nodes.

In this criterion, α is a parameter close to, but greater than one. Note that this criterion is coincident with the alpha shape concept [29,30].

Once a decision has been made concerning which of the nodes are on the boundaries, and which polyhedra are in the domain, the boundary surface must be defined. It is well known that, in 3D problems, the surface fitting a number of nodes is not unique. For instance, four boundary nodes on the same sphere may define two different boundary surfaces, one concave and the other convex.

In order to avoid this undefined boundary, the boundary surfaces will be defined with:

All the polyhedral surfaces having all their nodes on the boundary and belonging to just one polyhedron are boundary surfaces.

The boundary surface is necessary when the normals are needed in order to impose some boundary conditions. Furthermore, in weak formulations, the boundary surface is also important for a correct evaluation of the domain volume.

Nevertheless, it must be noted that in the criterion proposed above, the error in the boundary surface definition is proportional to h. This is the error order accepted in a numerical method for a given node distribution. The only way to obtain more accurate boundary surface definition is by decreasing h.

5. Element quality indicator

A polyhedron quality indicator will be defined in order to evaluate the partition obtained with the EDT compared with other methods.

A wrong mesh is a mesh which is not acceptable to be used in a numerical method. The main drawback of a Delaunay mesh is the presence of tiny volume polyhedra. In fact, the problem of tiny volumes is the existence of shape functions with very large gradients. These large gradients deteriorate the numerical solution.

Classical definition of mesh quality [3] will not be used here. The reason is that in this paper a node distribution is considered for which an optimal partition must be found without moving or removing nodes. The presence of nodes very close to each other in the node distribution is considered as a necessity of the physical problem to represent correctly a gradient in the solution and not as a wrong partition.

For instance, tetrahedra named *spires* and *wedges* in Fig. 6 will be considered elements of good quality because they are the best elements for a node distribution with the nearest nodes along one direction and more separated nodes in the other. These elements represent correctly the gradient "expected" in each direction to solve a particular physical problem. On the other hand, *slivers* or *splinters* as well as *caps*, will be considered wrong elements because they introduce shape functions with infinite or almost infinite gradients, which are not in agreement with the gradients expected for this node distribution.

In order to obtain a parameter to define which is a good or a wrong element to be used in a numerical method, but taking into account that the node distribution is fixed and introduced for physical needs, the gradient ratio γ will be defined as:

$$\gamma = \frac{\text{maximum gradient expected}}{\text{maximum gradient of the shape functions}}. \tag{1}$$

The maximum gradient expected for the node distribution will be defined in each polyhedron as $1/h_{\min}$, where $h_{\min}$ is the shortest distance between two nodes belonging to the polyhedron.

The maximum gradient of the shape functions may be evaluated directly using the largest gradient modulus from the shape functions of the polyhedron $|\nabla N_{\mathrm{p}}|_{\max}$ at the geometrical center. (See Appendix B for the definition of the shape functions N_{p}).

The quality of a polyhedron from a fixed node distribution is then numerically defined as:

$$\gamma = \frac{h_{\min}^{-1}}{|\nabla N_{\mathrm{p}}|_{\max}}. \tag{2}$$

It must be noted that in the EDT algorithm there is not any smoothing process in which element qualities must be evaluated many times. The gradient ratio γ is defined here in order to verify the quality of the resulting mesh, only for the sake of this paper. It is by no means used in the normal operation of the program.

The ideal element will therefore have a large gradient ratio. This element will have its nodes near the same Voronoï sphere and an acceptable shape function for numerical computations as previously required. On the other hand, bad-shaped elements have small gradient ratios. The smaller the ratio, the worse will indicate high shape function gradients capable to destroy the numerical solution.

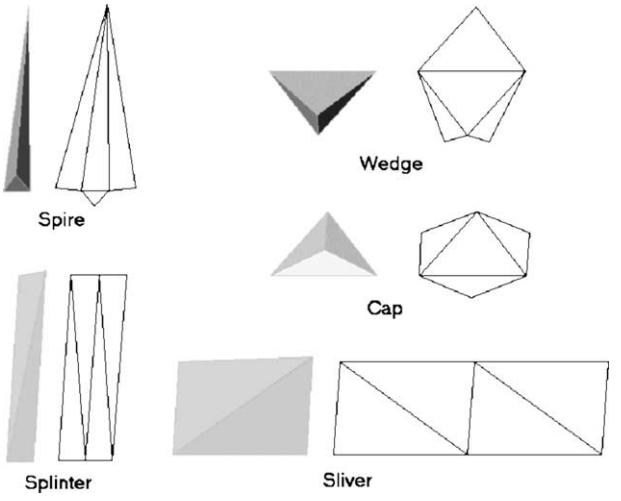

Fig. 6. Perspective, unfolding and naming of some bad-shaped tetrahedra.

Taking into account that the computer precision nowadays is of order 10^{-16}, and also the results from the numerical test performed in the next section, a gradient ratio $\gamma > 10^{-3}$ is recommended in order to accept a mesh for numerical computations.

6. Numerical tests

6.1. Solution of the Poisson equation on a cube

In order to check whether or not a mesh is acceptable to be used in a numerical method, a physical problem must be solved using that partition.

A cube of unit side, with an internal exponential source, has been used to validate the EDT.

The problem to be solved is the classical Poisson equation:

$$\nabla^2 u = f(x,y,z) \tag{3}$$

with the internal source:

$$\begin{aligned} f(x,y,z) = (&-2kyz(1-y)(1-z) + (kyz(1-y)(1-z)(1-2x)^2)^2 \\ &- 2kzx(1-z)(1-x) + (kzx(1-z)(1-x)(1-2y)^2)^2 \\ &- 2kxy(1-x)(1-y) + (kxy(1-x)(1-y)(1-2z)^2)^2) \\ &\times (-\mathrm{e}^{kxyz(1-x)(1-y)(1-z)}/(1-\mathrm{e}^{k/64})). \end{aligned} \tag{4}$$

The boundary condition is the unknown function u set equal to zero on all the boundaries.

This problem has the following analytical solution:

$$u(x,y,z) = (1-\mathrm{e}^{kxyz(1-x)(1-y)(1-z)})/(1-\mathrm{e}^{k/64}). \tag{5}$$

Several node distributions have been tested with 125 (5^3), 729 (9^3), 4913 (17^3), and 35 937 (33^3) nodes, with structured and non-structured node distributions. In all cases, the numerical solution was obtained using linear finite element shape functions for tetrahedral elements and polyhedral shape functions (as defined in Appendix B) for polyhedral elements.

The following procedure was used to generate the structured node distributions: initially all the nodes were in a regular cubic array with a constant edge length h. Then, each internal node has been randomly displaced a distance ρh (with $\rho \ll 1$) in order to have an arbitrary, but structured, node distribution. Surface and edge nodes are perturbed but remaining in the surface or the edge. Corner nodes were not perturbed. In this paper, the parameter ρ was fixed to 10^{-6}.

The 3D non-structured node distribution was generated using the GID pre/post-processing code [31] with a constant h distribution. GID generates the nodes using an AFT, which guarantees that the minimal distance between two nearby nodes lies between $0.707h$ and $1.414h$.

6.1.1. Extended Delaunay tessellation vs. standard Delaunay tessellation

It must be noted that in 2D problems, both node distributions generated as described before, structured and non-structured, will give a Delaunay partition with near-constant area triangles, which is optimal for a numerical solution. Nevertheless, this is not the case in 3D problems in which, even for a constant h node distribution, many zero or near-zero volume tetrahedra (slivers) will be obtained on a standard Delaunay partition. Fig. 7 shows, for instance, the presence of slivers on a structured eight-node distribution. The eight nodes have been randomly displaced a small distance from the regular cubic array in order to force the generation of wrong tetrahedra. Slivers may introduce large numerical errors in the

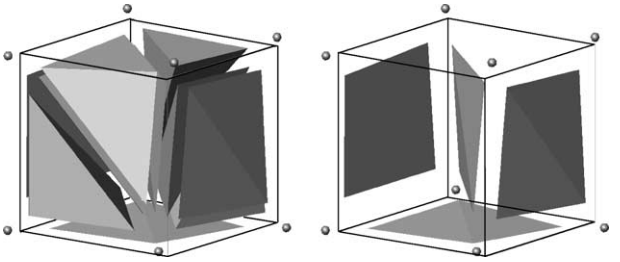

Fig. 7. Presence of slivers in a Delaunay partition of a perturbed cube. Left: tetrahedra produced by the Delaunay partition. Right: slivers isolated.

solution of the unknown functions and their derivatives, which may completely destroy the solution. In standard DT, slivers are eliminated by means of a cosmetic algorithm, like edge-face swapping or moving the nodes.

In order to show this behavior and to show that the EDT eliminates slivers automatically from the mesh, without any cosmetic algorithm, the following tests were performed:

For a fixed-node set (e.g. 17^3 nodes), δ was swept from 0 to 10^{-1}. Extremely low values of δ results in meshes equal or similar to the DT. Large values of δ allow more spheres to be joined, giving well-shaped polyhedra displaying the full power of the EDT.

Fig. 8 shows the error in L^2 norm for the derivative of the solution of the 3D problem stated in Eqs. (3) and (4). This has been done both for structured and non-structured distributions against the δ parameter. It can be shown that in both cases the errors are very large ($\sim 10^1$) for $\delta < 10^{-6}$ and very small ($\sim 10^{-2}$) for $\delta > 10^{-5}$. Larger values of δ do not change the results.

This example is very important because it is showing that, for a given node distribution, the standard Delaunay concept without cosmetic does not work. Mesh generators currently use edge-face swapping or another algorithms to overcome the presence of wrong elements. All those operations are iterative. The idea of joining similar spheres, solves this problem on a very simple way. The wrong tetrahedra are automatically joined to form polyhedra with optimal shapes.

The results of Fig. 8 show also that δ must be large compared with the computer precision (e.g. $\sim 10^{-5}$ for a computer precision of order 10^{-16}) but also means that δ is not a parameter to be adjusted in each example because the results do not change by setting δ larger than 10^{-4}.

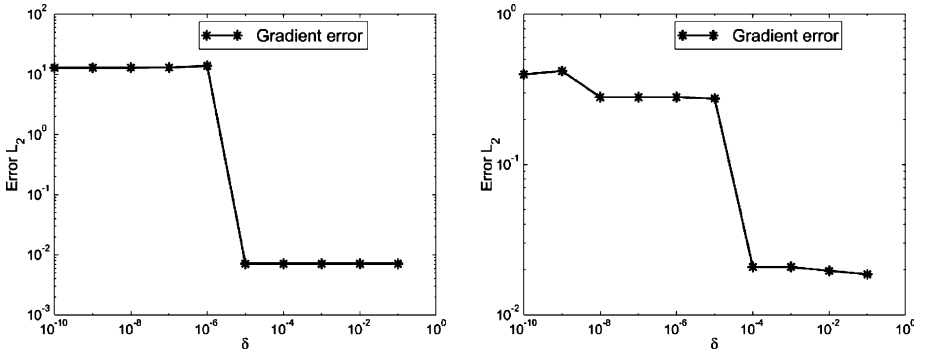

Fig. 8. Cube with exponential source. Error of the derivative in L^2. Left: structured node distribution. Right: non-structured node distribution.

6.1.2. Good and wrong elements

In order to show the performance of the EDT to generate good elements, the gradient ratios of the node distribution of the previous example was evaluated.

In Fig. 9, for each δ value, the heights of the columns represent the number of elements having their gradient ratios γ in the indicated range.

The importance of Fig. 9 is to show that, by joining similar spheres, the best polyhedra are automatically built. For the standard DT (and also for small δ values), there are bad tetrahedra (extremely low γ) for both the structured and the non-structured node distribution. By increasing δ the wrong elements disappear.

For the structured case, with an adequate δ value all the elements become "automatically" hexahedra (cubes). Note that the hexahedra are the best elements for this node distribution.

For the non-structured case, when δ is set to any value larger than 10^{-6}, EDT ensures a mesh without any element with γ lower than 10^{-3} (slivers). Note that values of δ greater than 10^{-1} gives the better elements, whose qualities γ are over 10^{-1}.

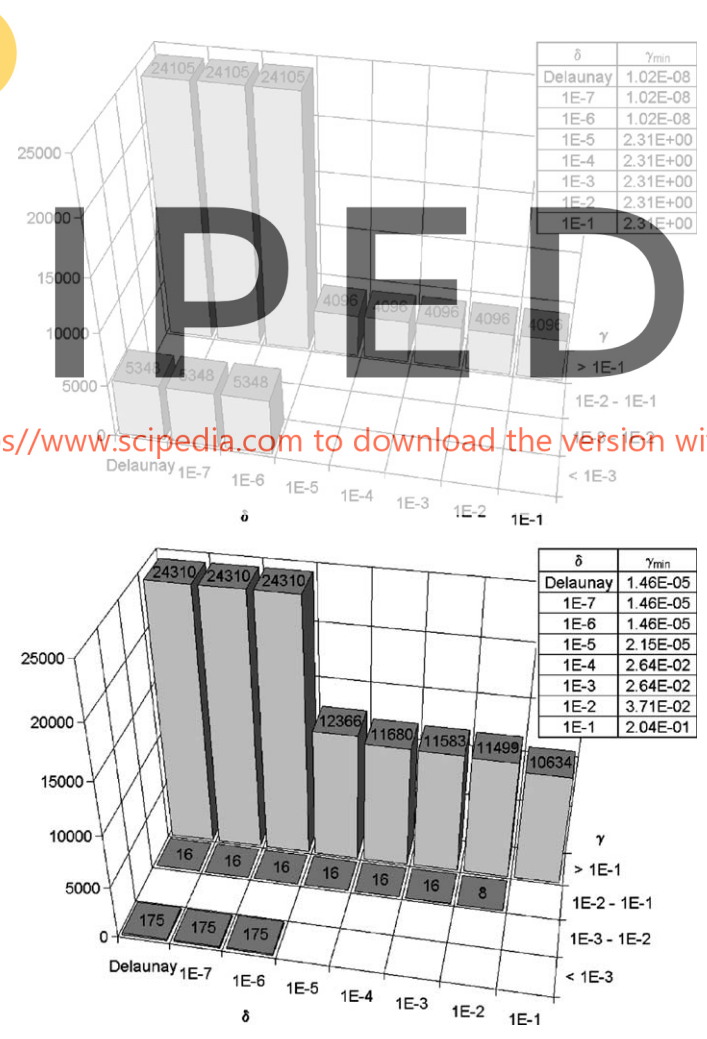

Fig. 9. Number of polyhedra vs. gradient ratio γ for different δ. Top: structured node distribution. Bottom: non-structured distribution made by GID.

All the examples shown in this paper were carried out with a fixed $\delta = 10^{-2}$. With that value, the resulting polyhedra have qualities $\gamma > 10^{-2}$.

6.1.3. Convergence rate

Fig. 10 shows the convergence of the above-defined example, when the number of nodes is increased from 5^3 to 33^3. The upper plots show the error in L^2-norm, both for the function and its derivatives. All the graphics show an excellent convergence rate. It must be noted that for all the non-structured node distributions tested (and also the structured ones for $\rho = 10^{-6}$), the FEM with elements generated using a DT gave totally wrong results, and even ill-conditioned matrices were frequently found during the stiffness matrix evaluation.

6.1.4. Computing times

In order to validate that the number of operation in the evaluation of the EDT partition is order near n, the computing time in a standard PC (Intel PIII 800 MHz) was analyzed. Fig. 11 shows the time in seconds for both the structured and non-structured cases.

A regression of the obtained results shows that the computing time is approximately:

$$t(\mathrm{s}) = 0.000325n^{1.08} \quad \text{for structured meshes and}$$

$$t(\mathrm{s}) = 0.000283n^{1.10} \quad \text{for non-structured,}$$

showing that the convergence exponent is close to 1 as expected in a DT.

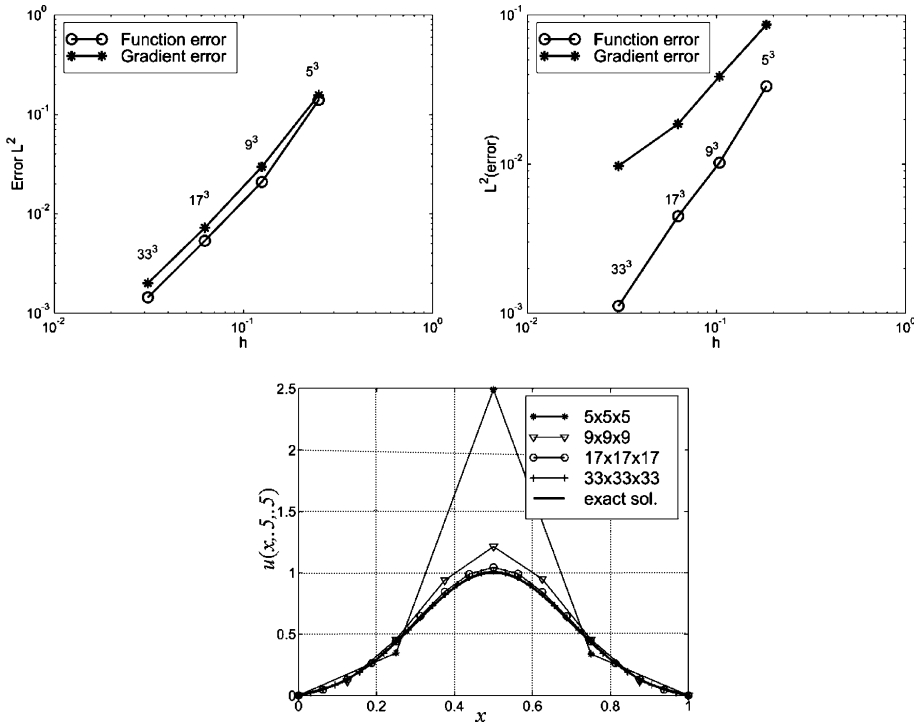

Fig. 10. Cube with exponential source. Convergence of the numerical solution and its gradient for different partitions. Upper left: structured node distribution. Upper right: non-structured node distribution. Below: centerline solutions obtained with structured node distribution (the same results were obtained using non-structured node distribution).

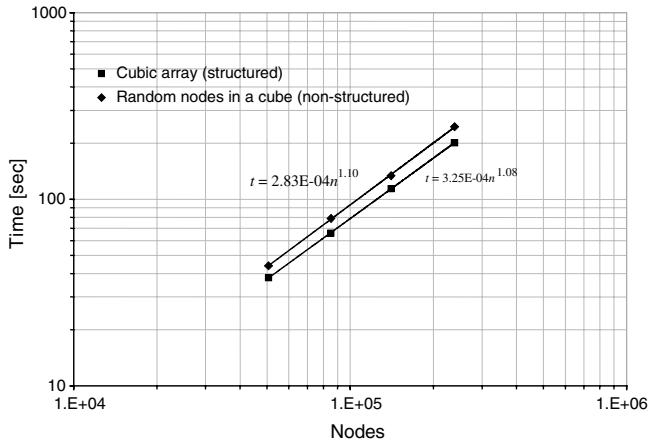

Fig. 11. Time vs. number of nodes in a standard PC.

6.2. Three-dimensional arbitrary geometry mesh

In order to show the performance of the method with in an arbitrary 3D geometry, a volume representing the vocal A was generated with distributions of $n \cong 10^3$ and 10^5 nodes, using the GID node generator with constant h. Fig. 12 left shows, for instance, the boundary node distribution for $n = 8452$ nodes.

Using the EDT algorithm described in Section 3 with the boundary surface definition described in Section 4 and an alpha shape parameter $\alpha = 1.3$, polyhedral partitions were found with external boundary surfaces represented in Fig. 12.

It must be noted the accurate definition of the boundary surface representing the letter A by the alpha shape method. The errors in the concavities are the intrinsic error for the h node distribution.

Table 1 summarizes the main characteristics of the polyhedral mesh made for the larger node set, compared with the tetrahedral mesh obtained via the standard DT.

From a total of 106,947 nodes, the standard DT generates 599,934 tetrahedra, but 349 of them are slivers. On the other hand, EDT generates 430,262 tetrahedra and 67,162 polyhedra with more than four nodes. None of them have gradient ratio smaller than 10^{-2}. Another interesting conclusion to take out from this example is that in the EDT mesh, 86.5% of the elements are tetrahedral and then, in these elements the definition of the shape functions (Appendix B) will be coincident with classical linear shape functions of the FEMs.

Fig. 12. Three-dimensional arbitrary geometry. Left: boundary nodes from the point distribution generated by GID with constant h and 8452 nodes. Center: boundary surface for 8452 nodes. Right: boundary surface for 106,947 nodes.

Table 1
Three-dimensional arbitrary geometry: comparison between EDT and DT

	EDT	DT
Nodes	106,947	106,947
Elements	497,424	599,934
Tetrahedra	430,262	599,934
Non-tetrahedra	67,162	0
Slivers	0	349
γ_{min}	1.84e−2	2.34e−16

Finally, Fig. 13 shows the slivers distribution in the mesh. Elements having quality γ smaller than 10^{-2} have been plotted in black. Zero bad polyhedra have been found in the partition given by the EDT.

6.3. Meshing tomographic data

The next example shows the problem of meshing the cranial bone of a Carnotaurus Sastrei dinosaur (Fig. 14).

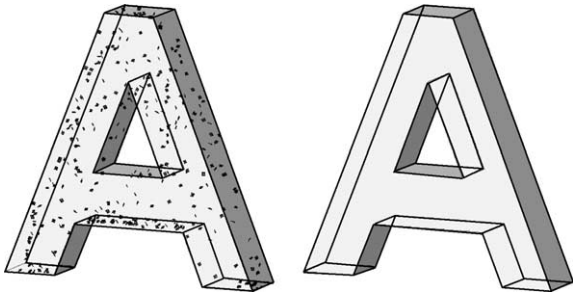

Fig. 13. Wrong elements in the 106,947 nodes tessellation. Left: 349 bad tetrahedral (slivers) in the DT. Right: zero wrong polyhedra in the EDT.

Fig. 14. Cranial bone of a Carnotaurus Sastrei dinosaur.

The node set was made starting with a stack of 2D tomographic images and then superimposing a regular array of points on each image and discarding the external nodes, resulting in a total of 110,676 nodes.

Fig. 15 shows the boundary surface obtained using $\alpha = 1.07$. The surface displays a great level of detail respecting the large variety of shapes present, with holes, cavities, and sharp concavities.

In order to show the advantages of the EDT over the standard DT, both meshes were generated. The standard DT gives a large amount of slivers whereas all the EDT elements are well shaped.

It must be noted that the node set is very structured, causing the presence of a lot of slivers in the standard DT. Fig. 16 shows, in the left, the DT slivers (the dark spots) positioned into the domain, at the right is shown that the EDT produces no slivers at all.

Table 2 summarizes the main characteristics of both meshes obtained.

In order remark the ability of the method for recognizing surfaces, a smooth-rendered image of the complete skull was performed using the obtained boundary surface (Fig. 17). The mandible has received the same processing as the cranial bone, and both pieces were mirrored to produce the displayed image.

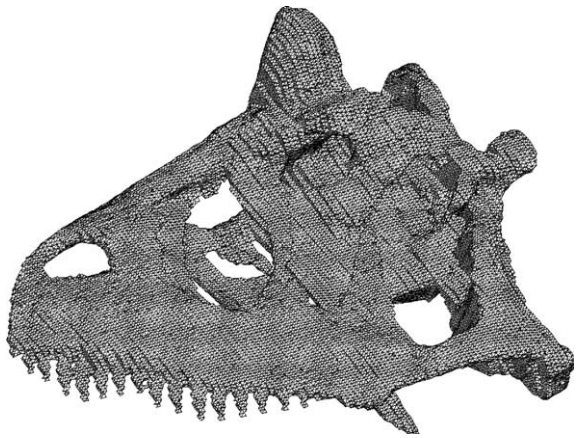

Fig. 15. Cranial bone: boundary surface.

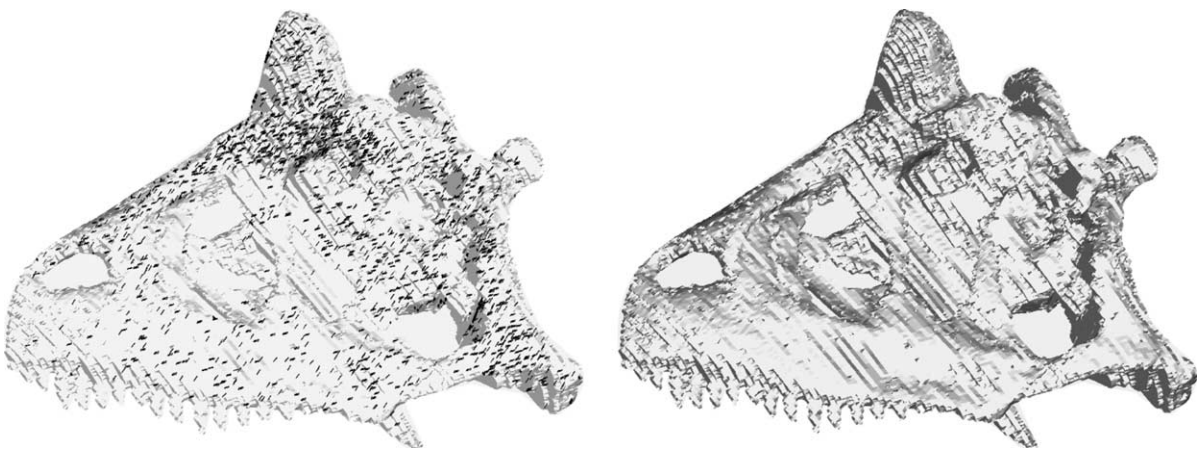

Fig. 16. Cranial bone. Left: sliver identification in the DT. Right: none sliver in the EDT.

Table 2
Cranial dinosaur bone: comparison between EDT and DT

	EDT	DT
Nodes	110,676	110,676
Elements	392,593	529,808
Tetrahedra	430,262	529,808
Non-tetrahedra	67,162	0
Slivers	0	1653
γ_{min}	2.55e−2	2.38e−7

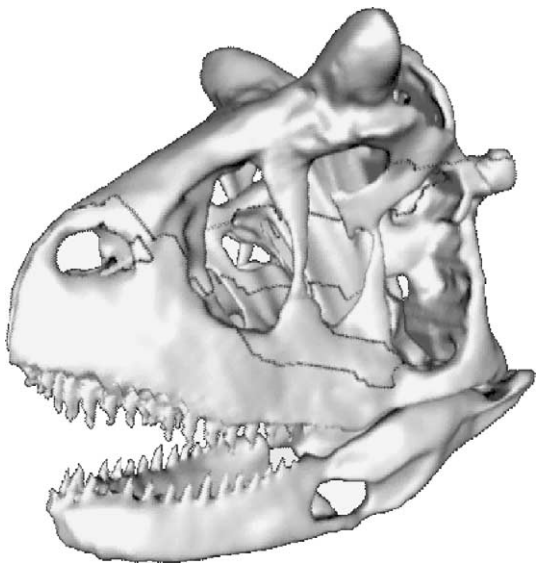

Fig. 17. Dinosaur skull: smooth-rendered image.

6.4. Solution of a fluid mechanic transient problem with free surface and breaking waves

The main motivation behind the development of the EDT method was to solve fluid mechanical problems using particle methods and Lagrangian formulations where a permanent mesh update and free-surface recognition are needed after each time step. In fact, the method has been successfully used in a large variety of such problems, both in 2D as well as in 3D domains [11].

The next example presents one of such problems solved: the collapse of a water column in a recipient.

This problem was experimentally and numerically solved in Ref. [6], and became a classical example for validating Lagrangian formulations.

Initially the column is located on one side of the recipient, supported by a removable board. The collapse starts when the board is slipped-up. The water suddenly falls running by the bottom until it impinges the opposite wall. The water goes up and then come down making a breaking wave and a set of minor travelling waves. The recipient walls were modeled by means of a set of fixed points and the initial water column by using a regular array.

Fig. 18 shows six time steps along with their corresponding meshes. Small circles represent boundary nodes recognized by the method.

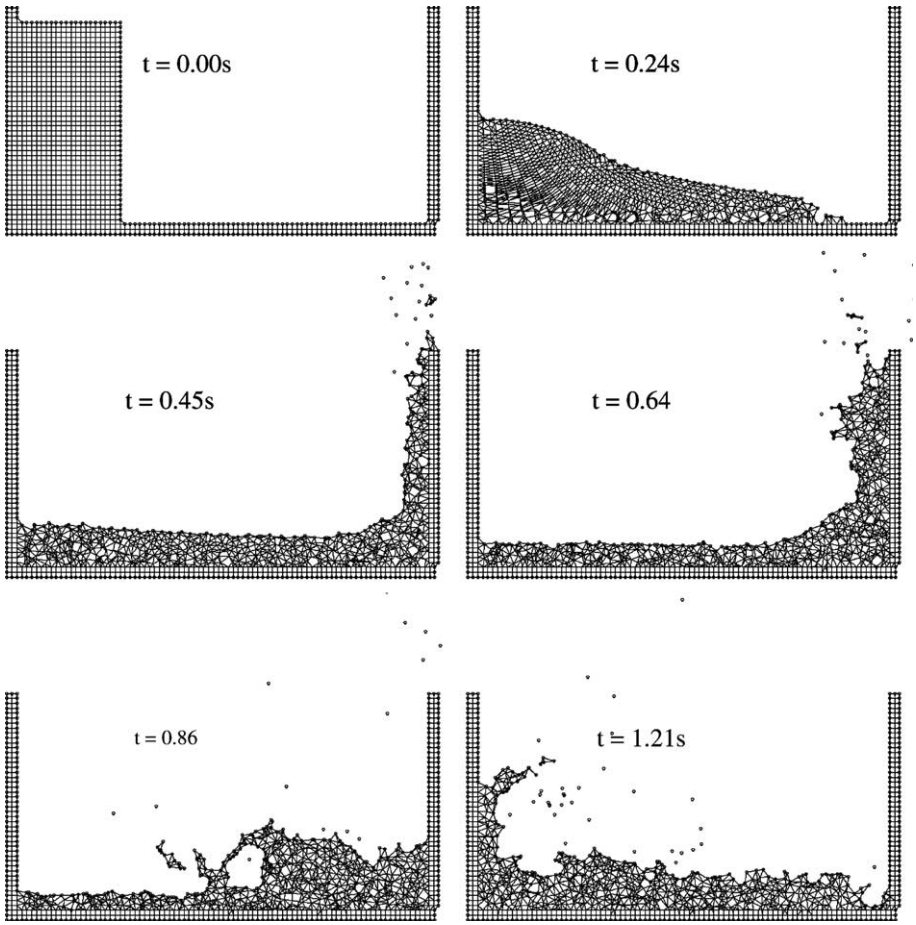

Fig. 18. Fluid mechanic transient problem: meshes and boundary nodes.

At time $t = 0$, almost all the elements were recognized as squares. Although on the following snapshots most of the elements are triangular, several non-triangular polygons were also found inside the domain. There are also some internal boundaries representing voids, which are related with the alpha shape recognition (see for instance $t = 0.86$).

Isolated nodes and small groups of isolated elements were found as disjoint pieces by the alpha-shape boundary-surface recognition. They represent small flying drops detached from the fluid.

The numerical results obtained, both in the timing (velocity of waves) and in the shape of the free surface, are in excellent agreement with the experimental result presented in Ref. [6].

7. Conclusions

For a given set of nodes with arbitrary 3D distribution and non-structured and variable distance between nodes, the method proposed gives a polyhedral mesh and a boundary surface with the following characteristics:

(a) The solution is unique for a given set of parameters δ and α. The solution is not sensitive to small variations of those parameters.
(b) Each polyhedron has all its nodes very close to the same empty sphere (optimal distance between nodes).
(c) Each polyhedron has an acceptable positive volume to be used in a numerical method. (The maximum gradient of the shape functions is not very large compared with the expected gradient for a given node distribution.)
(d) The correct definition of the boundary surface obtained depends on the local node distance between nodes $h(\mathbf{x})$.
(e) The computing time to achieve the mesh is of order near n, the same as the standard DT without further cosmetics.
(f) The large majority of the polyhedral elements generated are tetrahedra (around 85% in an arbitrary node distribution). This allows the use of standard linear finite element shape functions in all of them.

Acknowledgements

Thanks are given to Facundo Del Pin, for the results of the Poisson equation on the cube and for the fluid mechanical problem using the Lagrangian formulation. Thanks are also given to Adrián Cisilino and Gerardo Mazzetta for providing the data for the dinosaur example.

Appendix A. Criterion to join polyhedra

Consider two Voronoï spheres having nearby centers. See Fig. 19 for a two-dimensional reference.
As both Voronoï spheres are empty, they must satisfy the following relationship:

$$|r_2 - r_1| \leqslant \|\mathbf{c}_1 - \mathbf{c}_2\|, \tag{A.1}$$

where r are the radii and $\mathbf{c}$ the centers of the spheres.
Thus, two spheres are similar when their centers satisfy:

$$\|\mathbf{c}_1 - \mathbf{c}_2\| < \delta r_{\text{rms}}, \tag{A.2}$$

where δ is a small non-dimensional value and r_{rms} is the root-mean-square radius.
Two polyhedra will be joined if they belong to similar spheres. When all the nodes of a polyhedron belongs to another polyhedron, only the last one is considered.

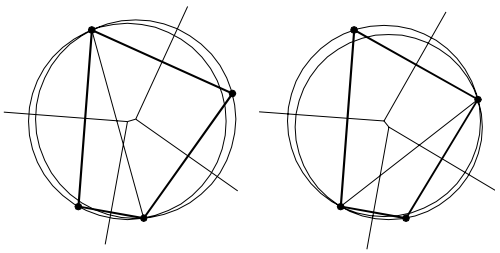

Fig. 19. Four nodes in near-degenerate position showing the empty circumcircles, the Voronoï diagram and the corresponding discontinuous Delaunay triangulation.

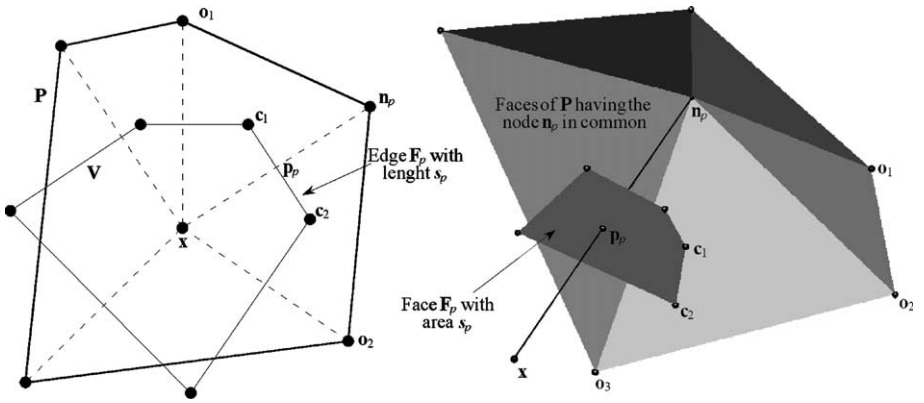

Fig. 20. Elements defining 2D and 3D shape functions.

Then, the algorithm consist in

(1) Find all the four-node empty spheres (DT).
(2) Join successively all the polyhedra using the above criterion.
(3) Delete all the polyhedra whose nodes belongs to another polyhedra.

Appendix B. Shape functions for arbitrary 3D polyhedra

For any point within a polyhedron $\mathbf{P}$, there is a Voronoï cell $\mathbf{V}(\mathbf{x})$ associated to the variable point $\mathbf{x}$ in the Voronoï tessellation of the set $\mathbf{P} \cup \{\mathbf{x}\}$.

Fig. 20 shows that every node $\mathbf{n}_\mathrm{p} \in \mathbf{P}$ has a corresponding face $\mathbf{F}_\mathbf{p}$ of $\mathbf{V}$, which is normal to the segment $\{\mathbf{x}, \mathbf{n}_\mathrm{p}\}$ by its midpoint. This is because $\mathbf{V}$ is the set of points closer than $\mathbf{x}$ than any other point.

Defining the functions:

$$\phi_\mathrm{p}(\mathbf{x}) = s_\mathrm{p}/\|\mathbf{n}_\mathrm{p} - \mathbf{x}\| = s_\mathrm{p}/h_\mathrm{p}, \tag{B.1}$$

as the quotient of the Lebesgue measure s_p of $\mathbf{F}_\mathbf{p}$ and the distance h_p between the point $\mathbf{x}$ and the node $\mathbf{n}_\mathrm{p}$. The shape functions are:

$$N_\mathrm{p} = \phi_\mathrm{p}/\Sigma_\mathrm{q}\phi_\mathrm{q}, \tag{B.2}$$

All the properties of this shape function can be found in Ref. [28].

References

[1] R. Lohner, Progress in grid generation via the advancing front technique, Engrg. Comput. 12 (1996) 186–210.
[2] P.L. George, H. Borouchaki, Delaunay Triangulation and Meshing: Application to Finite Elements, Hermes Sciences Publications, 1998.
[3] J.R. Shewchuk, What is a good linear element? Interpolation, conditioning, and quality measures, in: Eleventh International Meshing Roundtable, Ithaca, New York, Sandia National Laboratories, September 2002, pp. 115–126.
[4] N.A. Calvo, S.R. Idelsohn, All-hexahedral mesh smoothing with a node based measure of quality, Int. J. Numer. Methods Engrg. 50 (2001) 1957–1967.
[5] J.R. Shewchuk, Constrained Delaunay tetrahedralizations and provably good boundary recovery, in: Eleventh International Meshing Roundtable, Ithaca, New York, Sandia National Laboratories, September 2002, pp. 193–204.
[6] S. Koshizuka, Y. Oka, Moving particle semi-implicit method for fragmentation of incompressible fluid, Nucl. Engrg. Sci. 123 (1996) 421–434.

[7] R.A. Gingold, J.J. Monaghan, Smoothed particle hydrodynamics, theory and application to non-spherical stars, Mon. Not. R. Astron. Soc. 181 (1997) 375–389.
[8] G.A. Dilts, Moving least squares particle hydrodynamics. I Consistency and stability, Int. J. Numer. Methods Engrg. 44 (1999) 1115–1155.
[9] J. Bonet, S. Kulasegaram, Convection and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulation, Int. J. Numer. Methods Engrg. (1999).
[10] S.R. Idelsohn, M.A. Storti, E. Oñate, Lagrangian formulations to solve free surface incompressible inviscid fluid flows, Comput. Methods Appl. Mech. Engrg. 191 (2001) 583–593.
[11] S.R. Idelsohn, E. Oñate, F. Del Pin, A Lagrangian meshless finite element method applied to fluid–structure interaction problems, Comput. Struct. 81 (8–11) (2003) 665–671.
[12] A.P. Cisilino, G.V. Mazzetta, N.A. y Calvo, Reconstrucción geométrica y discretización con elementos finitos del complejo cráneo-mandibular del dinosaurio Carnotaurus Sastrei, Mecánica Comput. XXI (2002).
[13] E. Oñate, S.R. Idelsohn, O.C. Zienkievicz, R.L. Taylor, A finite point method in computational mechanics. Applications to convective transport and fluid flow, Int. J. Numer. Methods Engrg. 39 (22) (1996) 3839–3886.
[14] E. Oñate, S.R. Idelsohn, O.C. Zienkievicz, R.L. Taylor, C. Sacco, A stabilized finite point method for analysis of fluid mechanics problems, Comput. Methods Appl. Mech. Engrg. 39 (1996) 315–346.
[15] B. Nayroles, G. Touzot, P. Villon, Generalizing the finite element method: diffuse approximation and diffuse elements, Comput. Mech. 10 (1992) 307–318.
[16] W.K. Liu, S. Jun, S. Li, J. Adee, T. Belytschko, Reproducing kernel particle methods for structural dynamics, Int. J. Numer. Methods Engrg. 38 (1995) 1655–1679.
[17] N.R. Aluru, G. Li, Finite cloud method: a true meshless technique based on a fixed reproducing kernel approximation, Int. J. Numer. Methods Engrg. 50 (10) (2001) 2373–2410.
[18] N. Sukumar, B. Moran, A.Yu. Semenov, V.V. Belikov, Natural neighbor Galerkin methods, Int. J. Numer. Methods Engrg. 50 (2001) 1–27.
[19] A. Bowyer, Computing Dirichlet tessellations, Comput. J. 24 (2) (1981) 162–166.
[20] D.F. Watson, Computing the *n*-dimensional Delaunay tessellation with application to Voronoi polytopes, Comput. J. 24 (2) (1981) 167–172.
[21] J.R. Shewchuk, Lecture notes on Delaunay mesh generation. Available from <citeseer.nj.nec.com/shewchuk99lecture.html>.
[22] M. Bern, P. Plassmann, Mesh Generation in Handbook of Computational Geometry, Elsevier Science, 2000.
[23] H. Edelsbrunner, Damrong Guoy, An experimental study of sliver exudation, in: Proceedings, 10th International Meshing Roundtable, Sandia National Laboratories, October 7–10, 2001, pp. 307–316.
[24] J.R. Shewchuk, Adaptive precision floating-point arithmetic and fast robust geometric predicates, Discr. Comput. Geomet. 18 (1997) 305–363.
[25] S. Schirra, Precision and robustness issues in geometric computation, in: Handbook on Computational Geometry, Elsevier Science Publishers, Amsterdam, The Netherlands, 1999.
[26] N. Calvo, S.R. Idelsohn, All-hexahedral element meshing: generation of the dual mesh by recurrent subdivision, Comput. Methods Appl. Mech. Engrg. 182 (3–4) (2000) 371–378.
[27] V. Belikov, A. Semenov, Non-Sibsonian interpolation on arbitrary system of points in Euclidean space and adaptive generating isolines algorithm, in: Numerical Grid Generation in Computational Field Simulation, Proceedings of the 6th International Conference, Greenwich University, July 1998.
[28] S.R. Idelsohn, E. Oñate, N. Calvo, F. Del Pin, The meshless finite element method, Int. J. Numer. Methods Engrg, in press.
[29] H. Edelsbrunner, E.P. Mucke, Three-dimensional alpha shapes, ACM Trans. Graphics 13 (1994) 43–72.
[30] N. Akkiraju, H. Edelsbrunner, M. Facello, P. Fu, E. Mücke, C. Varela, Alpha shapes: definition and software, in: Proceedings of the 1st International Computational Geometry Software Workshop 1995, pp. 63–66. Available from <http://www.geom.umn.edu/software/cglist/GeomDir/shapes95def/>.
[31] GID The personal pre and post processor, CIMNE, Barcelona, 2002. Available from <http://gid.cimne.upc.es>.